REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
15-10-2009	Final Report	23 May 2008 - 23-May-09

4. TITLE AND SUBTITLE

Better Steganalysis (BEST) - Reduction of Interfering Influence of Image Content on Steganalysis Phase 2

5a. CONTRACT NUMBER
FA8655-08-1-3059

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Dr. Andreas Westfeld

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Technische Universität Dresden
Oskar-Seyffert-Str. 34
Dresden 01189
Germany

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
Unit 4515 BOX 14
APO AE 09421

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Grant 08-3059

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Steganography is the art and science of invisible communication. Its aim is the transmission of information embedded invisibly into carrier data. The goal of steganalysis is to discover steganographic alterations to carrier data. The project BEST (Better Steganalysis) explored the influence of image content when detecting steganography. If this influence, which interferes with the detector statistics, is reduced, the reliability of steganalysis can be increased. Steganalysis can be regarded as an attempt to separate image (or carrier) content from embedded steganographic signals. This project consisted of two phases. During its first phase, a new technique to scan elements of potential steganograms allowed a more fine-grained evaluation of the group size, a parameter of the RS attack by Fridrich et al., which affects the detection performance. The results of Phase 2 include newly developed attacks for advanced JPEG steganography like MB2 that use targeted features to blockiness reduction artefacts. Phase 2 also comprises a new generic methodology to apply spatial domain attacks in the DCT domain. All attacks are implemented in a software package called steganographic workbench (swb), which is described in this report. The package also includes implementations of some previous state of the art attacks that have been used for evaluation.

15. SUBJECT TERMS

EOARD, Information Theory, Steganalysis

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18, NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			JAMES LAWTON Ph. D.
UNCLAS	UNCLAS	UNCLAS	UL	49	19b. TELEPHONE NUMBER *(Include area code)* +44 (0)1895 616187

TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science, Institute of Systems Architecture

Better Steganalysis (BEST) Final Report

Reduction of Interfering Influence of Image Content on Steganalysis *

Prof. Dr.-Ing. Andreas Westfeld

andreas.westfeld@htw-dresden.de

October 8, 2009

*This project was supported by the Air Force Office of Scientific Research under the research grant numbers FA8655-06-1-3046 (BEST) and FA8655-08-1-3059 (BEST2). The U.S. Government is authorised to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Air Force Office of Scientific Research, or the U.S. Government.

Contents

List of Figures

List of Tables

Summary

Steganography is the art and science of invisible communication. Its aim is the transmission of information embedded invisibly into carrier data. The goal of steganalysis is to discover steganographic alterations to carrier data.

The project BEST (Better Steganalysis) explored the influence of image content when detecting steganography. If this influence, which interferes with the detector statistics, is reduced, the reliability of steganalysis can be increased. Steganalysis can be regarded as an attempt to separate image (or carrier) content from embedded steganographic signals.

This project consisted of two phases. During its first phase, a new technique to scan elements of potential steganograms allowed a more fine-grained evaluation of the group size, a parameter of the RS attack by Fridrich et al., which affects the detection performance.

The results of Phase 2 include newly developed attacks for advanced JPEG steganography like MB2 that use targeted features to blockiness reduction artefacts. Phase 2 also comprises a new generic methodology to apply spatial domain attacks in the DCT domain.

All attacks are implemented in a software package called *steganographic workbench* (swb), which is described in this report. The package also includes implementations of some previous state of the art attacks that have been used for evaluation.

Chapter 1

Introduction

Steganography is the ancient art and young science of hiding the communication in such a way that the existence of the private message cannot be detected. It works by hiding the secret message in a harmless carrier object. While cryptography merely ensures the confidentiality of the message content, steganography adds another layer of secrecy by concealing the fact that secret communication takes place.

Progress in steganography is tightly coupled with improvements of steganalysis, the detection of hidden information. In this report we introduce several techniques that aim to reduce the interfering influence of image content on steganalysis:

- optimisation of the group size parameter for RS steganalysis,
- separation of error sources for measurement of the influence of image content,
- a generic methodology to apply spatial domain attacks to frequency domain,
- two new more reliable higher order statistical attacks on randomised Jsteg, one for small and one for large images, and
- a powerful attack on MB2 using targeted features to blockiness reduction artefacts.

Chapter 2

Methods, Assumptions, and Procedures

2.1 A Steganographic Workbench

This report is intended to present the key ideas supported by the software in practical examples.

The software of this project, the *steganographic workbench* or `swb` package is implemented in R. Some parts are optimised for performance reasons in C.

R is a free programming language and computing environment that has become a de-facto standard among statisticians for the development of statistical software. Despite its strong points, and its large user base among statisticians, it is fair to say that R and its commercial cousin S-PLUS are not widely used in the signal processing community.

As far as our experience goes, R can significantly lower the threshold for experimental prototype implementations. While nested loops are necessary to process image files in C or C++, it is just one line of code in R due to the compact, clear and math-like syntax. Apart from the time saved while writing and reading the code, one source of its errors—the complexity—is suppressed.

To run the "steganographic workbench," an installed version of R is necessary (www.r-project.org). To install the `swb` package on Windows, start the R GUI environment and select "Install package(s) from local zip file. . . " in the "Packages" menu. Then choose the location of `swb.zip` and confirm.

On Unix systems this is accomplished by the shell command line

```
bash# R CMD INSTALL swb
```

where the package file `swb.tgz` must reside in the current working directory. You need supervisor rights to make packages available to all users. If you don't have these rights or want to install it only in your home directory, create and specify your own target for R libraries first:

```
bash$ export R_LIBS=~/.R
```

```
bash$ mkdir -p $R_LIBS
bash$ R CMD INSTALL swb
```

Once the `swb` package is installed, the user can load it into R using the command `library(swb)`. This makes a the following collection of functions available:

I/O

Read or write image data from/to files.

read.gif[1]
read.jpg
read.pgm[1]
read.png
read.pnm[1]
write.png

Attacks/Feature Sets

Predict the use of steganography, estimate the length of the embedded message, or extract a feature vector in order to train a classifier.

abs.moments.attack
ca.attacks[1]
ca.estimate
ca.jphide[1]
ca.jsteg[1]
eca[1]
eca.jphide[1]
eca.jsteg[1]
gca.jphide[1]
gca.jsteg[1]
gca.new.jsteg[1]
f5attack[1]
f5estimate[1]
hcfcom.attack
jphide.normalise[1]
jsteg.normalise
jpairs.attack
jrs.attack
jspa.attack
jws.attack
mb1.esorics.attack[1]
mle.estimate
pairs.attack
rs.attack
spa.attack
qem
qem2
wb.attack
yu.attack[1]
yu.estimate
zp.attack[1]
zp.estimate
f23[1]
f274[1]
f324[1]
ext.dct.features[1]
ext.markov.features[1]
merged.features[1]

ROC Curve Handling

Assess the quality (power/ accuracy) of attacks or plot an ROC curve.

assess.roc[1]
eer[1]
farid[1]
fpr.thresholds[1]
ker[1]
plot.roc[1]
prepare.roc
rho[1]
thresholds[1]
tpr.thresholds[1]

Embedding/Extraction

Simulate the embedding of a message.

jphide.embed
jsteg.embed
lsb.embed
lsb.extract
ltsb.embed
ltsb.extract
pm1.embed
pm1.extract

Transformations

DCT, quantisation, and scanning along a plane filling curve.

ndctq[1]
nidctq[1]
jndctq[1]
jnidctq[1]
scanpath
plot.scanpath

Integer Operators

Most of these operators are available in R, but require time-consuming conversions to and from floating point format. For performance reasons we implemented integer versions in C.

iabs

[1]There is currently no man page for functions that are rather for internal use in the `swb` package.

iand	iterative.mean.int[1]	tbl
iasr	iterative.mean[1]	jcalibrate[1]
ilsr	ixor	jhist
ior	ctbl[1]	jnzcoeff[1]
irange	dtbl[1]	
isign	mtbl[1]	

For some functions there is an online help available. For instance, the user can display the man page of read.jpg by entering `?read.jpg` into R.

2.2 Recursive Scanpaths

Many tools that detect steganographic payload extract a sequence of elements from the carrier media (e.g., images). In most cases the ordering of this sequence was just proposed by the ordering of the elements in the file. However, other sequences can lead to better detection performance. In addition we can turn a two-dimensional object like an image into a one-dimensional sequence object. Thus we can design detectors independently of an object's dimensionality.

Discrete objects of arbitrary complexity can be recursively scanned, if the recursion depth is chosen appropriately. We used recursive scanpaths to improve the detection power of the Pairs attack in images [23]. A space filling curve is a continuous map of a two-dimensional area (plane-filling) or a three-dimensional volume into a one-dimensional interval. David Hilbert, a German mathematician, invented a simple space-filling curve known as the Hilbert curve, which fills a square. The Hilbert curve can be encoded with the initial string L and the following string rewriting rules [20].

$$\begin{aligned} L &\longrightarrow +RF - LFL - FR + \\ R &\longrightarrow -LF + RFR + FL - \\ F &\longrightarrow \text{go one pixel forward} \\ + &\longrightarrow \text{turn right} \\ - &\longrightarrow \text{turn left} \end{aligned}$$

These rules terminate after a specific recursion depth.

Figure 2.1 shows the well-known Hilbert curve for the recursion depths 1, 2, and 3. The bold enumeration follows the curve and the italic follows the pixels in the file, which are stored row by row. The `swb` package provides a function `scanpath` that accepts the parameters width x and height y of the scanned image. The recursion depth d is internally determined for a Hilbert curve that is large enough to cover all pixels in the image:

$$d = \lceil \log_2 \max(x, y) \rceil \tag{2.1}$$

In most cases the Hilbert curve is able to cover more pixels than the actual image contains. The result of `scanpath` is a vector of $x \cdot y$ indices[2], which are used to

[2]The return type is actually a matrix that is used as an index vector. The two dimensions given as parameters to the function are stored as a dimension attribute for convenience.

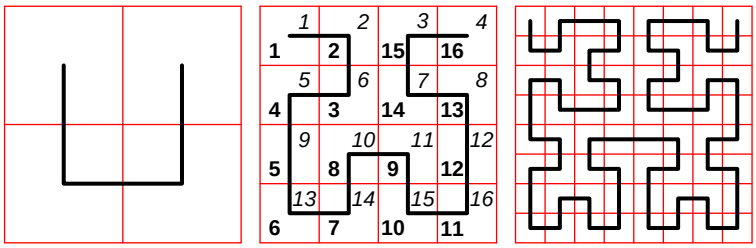


Figure 2.1: Hilbert curves for the recursion depths 1, 2, and 3

permute the pixels.

```
Example
lena <- read.png("lena.png")
sp <- scanpath(lena)
# or: sp<-scanpath(dim(lena)[1], dim(lena)[2], "hilbert1")
permuted.lena <- lena[sp]
```

In the permuted sequence of the image (i.e., pixels scanned along the Hilbert curve) almost all pixels are direct neighbours. This is also true if scanned row by row. However, also second order neighbours (every other pixel in the sequence) have a small mean distance (about $\sqrt{2}$), while this distance is about 2 in the original row-by-row order in the file. As a consequence, close pixels in the permuted sequence show a stronger correlation than before. This property was used in the aforementioned improvement of the Pairs attack.

The `swb` package implements further scan methods, `"hilbert1"` is only the default. The remaining are listed in the online help (enter `?scanpath` in the R command line). An example for a scan method can be displayed using the `plot.scanpath` method (e.g., `plot.scanpath("zigzag")`).

2.3 RS Group Sizes

RS analysis, which is another attack on LSB steganography, computes statistics on small disjoint groups of adjacent pixels[8]. Here it became evident that groups of pixels arranged in a square of $m \times m$ deliver better results than slices of $n \times 1$ pixels [8, 13]. This can be explained by the fact that locally close pixels differ less in colour or brightness than more distant ones. Certain types of systematically added randomness (e.g., LSB replaced by steganographic noise) are more easily detected in groups of closely related pixels with low variance in brightness. Unfortunately the quadratic shape scales in broad steps (group size $g = 1, 4, 9, \ldots$; see Fig. 2.2), which might impede the group size to be adapted to the image size in an optimal way. Scanning pixels along a recursive scan path will allow us to refine the step size and close the gap for the groups of pixels. Therefore it could improve RS if the optimal is in the middle of one of the previous coarse steps in group size.

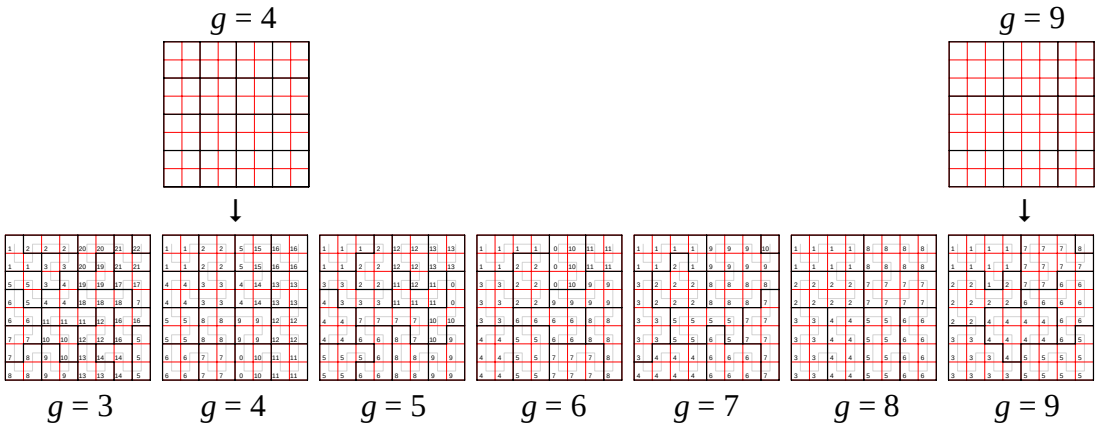


Figure 2.2: Conventional square shape of groups (above) compared to finer step size with proposed group shapes (below)

Along with refining the group size, recursive scanning can sequence elements: The pixels are no longer shaped in two dimensions. Therefore RS can work with slices of n pixels. In our experiments we consider all possible masks with $n = 2 \ldots 10$. However, we cannot expect different results for isomorphic masks, e.g. for the mirrored masks 00101 and 10100. Likewise with the mirrored version of an image: RS will give a similar result for it. Not so for mask 01010. Although the bits are just shifted by one position, it is different if the bit to be flipped (1 in the mask) is at the end of the mask (influencing only one neighbour difference in the noise measure) or in the middle (influencing two differences). The rules to reproduce this set are:

1. include the binary pattern of ascending unsigned integers with $n = 2 \ldots 10$ bits unless a subsequent condition is fulfilled,
2. exclude masks where all bits are the same, since they are unusable for RS,
3. exclude a mask if it is a mirrored version of a mask that is already included,
4. exclude a mask if it is a version of a previously included mask that was shifted to the left by two or more positions.

Table 2.1 shows an example for RS masks of length $n = 4$ that can be skipped in the experiments due to isomorphism. There are 776 different masks with $n = 2 \ldots 10$ bits; about one quarter of the workload is saved due to isomorphism.

2.4 Separation of Error Sources

In this project we proposed some methods that estimate the length of the embedded message. We will average length estimates in order to separate different sources of errors. Böhme and Ker defined three error-measures [1], which are evaluated in our experiments.

Table 2.1: Drop isomorphic RS masks

Mask	Drop-reason or "✓"
□□□□	all equal (unusable)
□□□■	✓
□□■□	✓
□□■■	✓
□■□□	shift □□□■ left by 2 or more
□■□■	✓
□■■□	✓
□■■■	✓
■□□□	shift □□□■ left by 2 or more
■□□■	✓
■□■□	mirrored □■□■
■□■■	✓
■■□□	shift □□■■ left by 2 or more
■■□■	mirrored ■□■■
■■■□	mirrored □■■■
■■■■	all equal (unusable)

A *cell* is a set of steganograms, produced when different messages of the same length are embedded into one particular carrier medium. To separate the error sources we measure the statistical dispersion (inter quartile range, IQR, difference between 0.75- and 0.25-quantile) and the central tendency (median, 0.5-quantile) of length estimations for each cell. From these two measures we derive three kinds of error (cf. Fig. 2.3):

1. The *within-image error* is the median of all cell IQRs. This kind of error is induced by the message and the secret key that is used for embedding. The distribution of the cell IQRs passed tests of normality.

2. The *between-image error* is the IQR of all cell medians. This is an image-specific error. The cell medians are rather Student-*t* distributed with $\nu = 1 \ldots 10$ degrees of freedom for the proposed set of attacks.

3. Finally, the *bias* is the median of the cell medians. This error can be pre-computed for a given source of images and a particular length estimating attack.

2.5 Aspects of High Performance Computing

The separation of error sources requires a large number of repetitions for each image in the test database. Apart from that, the experiments cover a number of independent

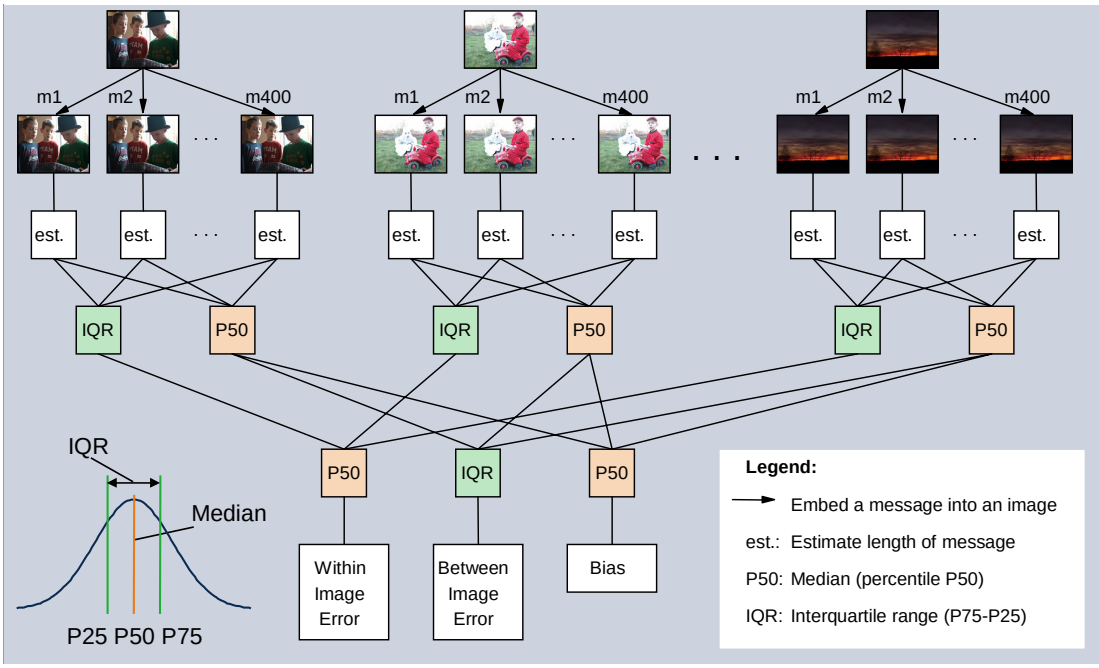


Figure 2.3: Separation of within-image error, between-image error, and bias

variations, such as five different image sizes (600×840, 400×560, 200×280, 80×112, and 40×56 pixels), with quality $q = 0.8$. The medium sized images (200×280) have been compressed at seven different qualities ($q = 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99$). We applied 8 different embedding rates (1 %, 5 %, 10 %, 20 %, 40 %, 60 %, 80 %, and 100 % of the steganographic capacity).

We repeat the embedding with 400 random messages for over 600 images, at 11 combinations of size and quality, and at 8 embedding rates, detecting the result with 78 attacks. This results in about 1700 million length estimations.

Since 630 cell IQRs seem to follow a Gaussian distribution, the application of the median will reduce the error by a factor $\sqrt{630}$ to a negligible level (4 %). The reduction for Student-t distributed values is slightly more efficient.

If the steps to load an image, embed, and estimate the length of the embedded message required about one second, a first naive estimation will expect an execution time of 54 years (472,000 hours). Hence the goal is (a) to combine as many steps as possible, but also (b) generate executions steps that are as small as possible to allow fair scheduling without pushing up the costs for loading the runtime environment.

Let t_l be the time to *load* an image into the system, t_e the time for *embedding* a random message, and t_a the time to execute an *attack* (length estimation). If we load an image only once for all related operations, and if we embed a message only once before applying the 78 attacks (cf. Figure 2.4), the necessary execution time is estimated as follows:[3]

$$t_\mathrm{total} = 11 \cdot 630 \cdot (t_\mathrm{l} + (8 \cdot 400 \cdot (t_\mathrm{e} + 78 \cdot t_\mathrm{a}) + 78 \cdot t_\mathrm{a}))$$

[3] It is not necessary to repeat the attack for embedding rate 0.

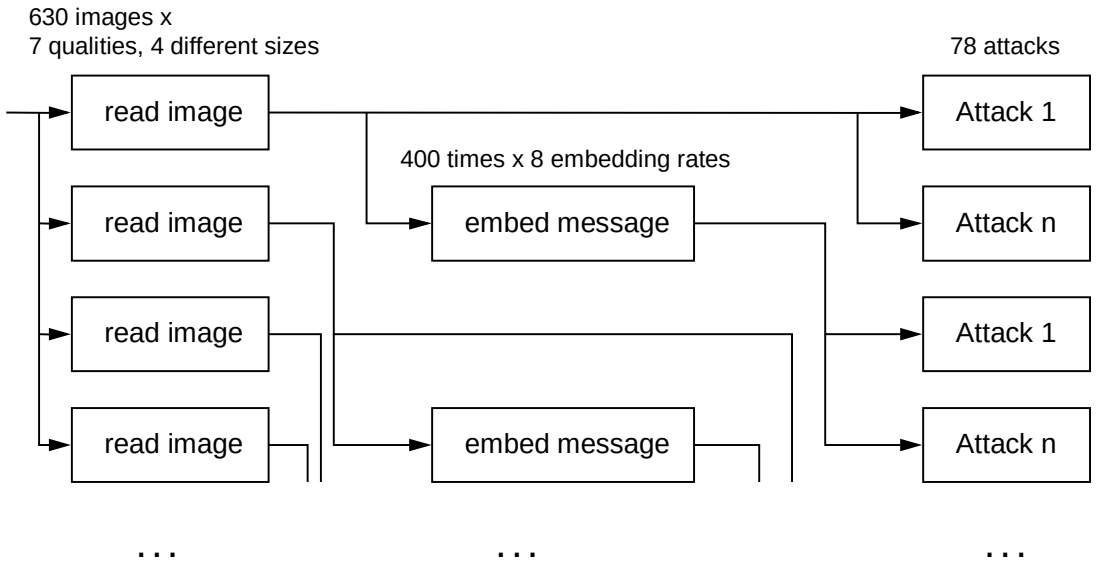


Figure 2.4: Pipeline for high performance computing

Different image sizes are weighted differently:

$$t_{\text{total}} = (1 + 0.48 + 7 \cdot 0.10 + 0.03 + 0.02) \cdot 630 \cdot (t_{\text{l}} + (8 \cdot 400 \cdot (t_{\text{e}} + 78 \cdot t_{\text{a}}) + 78 \cdot t_{\text{a}}))$$

With $t_{\text{l}} = 0.123\text{s}$, $t_{\text{e}} = 0.335\text{s}$, and $t_{\text{a}} = 0.0577\text{s}$, we expect an execution time of about 252 days. However, on a PC farm with about 2500 CPU cores at TU Dresden, this job was reduced to some hours wall time. Nevertheless, it was still challenging to aggregate and analyse all these results.

2.6 Applying Attacks from Spatial Domain in DCT Domain

During Phase 2 of BEST we applied spatial domain attacks in the DCT domain. In 2006 we published a paper on new histogram based attacks against Jsteg and JPhide (category attack). To be honest, we were not very confident about the contribution of this paper. Histogram based attacks have been used for a long time in the spatial domain. Such attacks (using first order statistics) have been outperformed in subsequent years by RS, SPA, WS. While the first histogram based attack (chi-square attack) has been used in both spatial domain *and* DCT domain, all the subsequent improvements were designed for the spatial domain only. RS, SPA, and WS were crying for a translation to the DCT domain. However, it was not obvious how to do so.

Apart from that, the chi-square attack can even be outperformed by other first order methods, e.g., dual statistics. Dual statistics have been ported to the DCT domain in the 2006 paper—a provoking result, since it was not only an improvement compared to the aged chi-square attack, but also outperforming contemporary blind attacks. This

result was followed by papers proposing blind attacks with an increased number of features that enabled a more reliable detection.

In this section we describe the functions in the `swb` package that have been used for the evaluation of spatial methods in the DCT domain.

2.6.1 JPEG Image Handling

All these methods will operate on DCT coefficients. The function that reads a JPEG file and creates an R structure is called read.jpg. Its implementation uses the libjpeg. The

```
require(swb)               # load swb library
j <- read.jpg("lena.jpg") # store data from lena.jpg in j
```

The structure of its return value is modelled on Phil Sallee's JPEG toolbox for Matlab.[4] `j` contains the losslessly decompressed DCT coefficients along with the quantisation tables and other meta-information.

```
> str(j)
List of 14
 $ width          : int 512
 $ height         : int 512
 $ icolor         : int 1
 $ ncomp          : int 1
 $ jcolor         : int 1
 $ njpegcomp      : int 1
 $ optimize.coding: logi FALSE
 $ comments       : int 1
 $ coefficients   :List of 1
  ..$ : int [1:64, 1:4096] 47 2 1 0 0 0 0 0 2 0 ...
 $ qtables        :List of 1
  ..$ : int [1:8, 1:8] 5 3 3 5 7 12 15 18 4 4 ...
 $ ac.huff.tables : int 1
 $ dc.huff.tables : int 1
 $ info           :List of 6
  ..$ component.id : int 1
  ..$ h.samp.factor: int 2
  ..$ v.samp.factor: int 2
  ..$ quant.tbl.no : int 1
  ..$ dc.tbl.no    : int 1
  ..$ ac.tbl.no    : int 1
 $ progressive    : int 0
```

[4] During this project we also implemented his MB1 and MB2 methods in R and consequently needed a compatible input function

The above image "Lena" has 512 by 512 pixels. The structure of this JPEG image in `j` can be displayed using R's built-in function `str`. For JPEG files, the image is decomposed into disjoint blocks of 8 $\times$ 8 pixels. Hence, the "Lena" image consists of 64$\times$64 = 4096 blocks, each of which is transformed using the DCT into 64 coefficients. Since this is a greyscale image, the file contains only one brightness channel. The coefficients of the first and only channel can be extracted by

```
coeffs <- j$coefficients[[1]]
```

2.6.2 Scanpaths

`coeffs` is arranged in a 64 $\times$ 4096 matrix. The first row contains all 4096 DC coefficients, rows 2...64 the AC coefficients. If we have a sorted image database with equally sized images, we can pre-calculate the index vectors for some scanpaths in advance:

```
d <- sqrt(dim(coeffs)[2])          # sqrt(4096)=64
HP <- scanpath(d, d)
HP2 <- scanpath(d, d, "hilbert2")
SL <- scanpath(d, d, "slalom")
```

There is also a function to embed a random message. The embedding method is Jsteg along a randomised embedding path. In the next example, we use 10 percent of the capacity (embedding rate 0.1) and store the resulting steganogram in `s`:

```
s <- jsteg.embed(coeffs, 0.1)
```

Before the set of steganalytic attacks is applied to the steganogram `s`, its elements can be reordered. If the matrix `s` is converted to a vector, the coefficients are in intra block order, i.e., the first 64 coefficients belong to the same 8 $\times$ 8 pixel block. Another option is the zigzag order that is used in the JPEG standard and easily achieved by permutation of the 64 columns in the matrix here. For this purpose the `swb` package has a global variable `zigzagorder` containing the appropriate permutation. R provides a function `t()` that transposes a matrix (turns rows into columns and vice versa). It can be used to switch from *intra* to *inter* block scan. In the transposed matrix, each 4096 elements belong to the same frequency, starting with the DC coefficients, and two consecutive coefficients belong to two different 8 $\times$ 8 pixel blocks.

```
x <- s                      intra block scan
x <- s[zigzagorder,]        zigzag order
x <- t(s)                   inter block scan row by row
```

In addition, the block ordering can be permuted using one of the pre-calculated index vectors for Hilbert 1 path `HP`, Hilbert 2 path `HP2`, or slalom `SL`.

```
x <- t(s[,HP])              inter block scan Hilbert 1 path
x <- t(s[,HP2])             inter block scan Hilbert 2 path
x <- t(s[,SL])              inter block scan slalom path
```

we can remove the DC coefficients from each of the aforementioned scanpaths. Although DC coefficients are affected by Jsteg, some steganalytic methods perform better if only AC coefficients are used for the attack [15]. The first column of a matrix is easily deleted in R by a -1 in the column index. (`s[-1,]` is equivalent to `s[2:64,1:4096]`, selecting all but the first column.)

```
x <- s[-1,]                 intra block scan (AC only)
x <- s[zigzagorder,][-1,]   zigzag order (AC only)
x <- t(s[-1,])              inter block scan row by row (AC only)
x <- t(s[-1,HP])            inter block scan Hilbert 1 path (AC only)
x <- t(s[-1,HP2])           inter block scan Hilbert 2 path (AC only)
x <- t(s[-1,SL])            inter block scan slalom path (AC only)
```

2.6.3 Histogram-Based Attacks

We will use three known histogram-based detection methods for randomised Jsteg to evaluate the reliability and precision of the methods that are proposed in the next subsection. The method of Zhang and Ping (ZP) divides the histogram into two interleaved groups [28], the attack of Yu et al. estimates the probability density function of the quantised values on a basis that is invariant to steganographic embedding [27]. The category attack (CA) compares equalising and complementary pairs of values with each other [15]. All three can estimate the length of the embedded message. For further details the reader is directed to the description in the respective original publications. As suggested in Section 2.5, reusable intermediate results are computed only once per image. Since the histogram is used in all three attacks, it is determined with a separate function `jhist`. The estimated embedding rate is stored in `p`.

```
hr <- jhist(x)
p <- zp.estimate(hr)
p <- yu.estimate(hr)
p <- ca.estimate(hr)
```

2.6.4 Proposed Attacks

While there have been several proposals to use higher order statistics in the spatial domain, no specific method existed to the best of our knowledge for the DCT domain. In the following we will apply several techniques that are known from the spatial domain to (selected) DCT coefficients, which have been scanned along different paths (intrablock and interblock).

In this subsection the spatial domain attacks RS, Pairs, SPA, and WS [8, 10, 4, 7] are adapted to DCT coefficients [25]. In most cases it was easy to make the upper input limit of the attacks (usually fixed at 255 for maximum brightness) a more general parameter. For further details the reader is directed to the respective original publications.

There are two basic requirements for the input if attacks have been designed for brightness or colour intensity values.

1. The input should be non-negative. (However, DCT coefficients are positive and negative integers.)
2. The input should not contain any values that are excluded from embedding. (However, the DCT values 0 and 1 are not used for embedding by randomised Jsteg.)

The function `jsteg.normalise` processes the sequence of coefficients to match the two requirements:

```
nc <- jsteg.normalise(x)
```

The modified attacks are applied directly to the normalised coefficients. An exception is the newly developed WB attack that also considers steganographically unused coefficients in its prediction. Hence it is applied to the coefficients in `x`. The JRS attack has a mask parameter. We used the two masks □■ and □■□ in our experiments. The functions for JRS, JPairs, JSPA, JWS, and WB are used as follows (the estimated length will be stored in `p`):

```
p <- jrs.attack(nc, c(0,1))
p <- jrs.attack(nc, c(0,1,1,0))
p <- jpairs.attack(nc)
p <- jspa.attack(nc)
p <- jws.attack(nc)
p <- wb.attack(x)
```

2.7 Attacks on Some Advanced Embedding Methods

2.7.1 Plus-Minus-One Embedding

One common embedding method is known as plus-minus-one embedding (PM1). Due to its simplicity it is not really "advanced." It is a rather simple derivative of the LSB replacement method. However, while the ubiquitous LSB embedding method can be detected with high reliability, PM1 requires advanced steganalytic methods to be detected.

To illustrate LSB embedding, let us consider greyscale images with pixel values in the range $0 \ldots 255$ as carrier medium. LSB steganography replaces the least significant bit of each pixel value in the image with the corresponding bit of the message to be hidden. An even-valued pixel will either retain its value or be incremented by one. However, it will never be decremented. The converse is true for odd-valued pixels. This asymmetry introduces a statistical anomaly in the histogram of brightness values: pairs of values (PoV) that consist of an even value and its successor tend to exhibit the same frequency if the image contains an LSB embedded message. This can be exploited for steganalytic purposes [26, 8, 4, 10, 7].

PM1, also known as LSB matching, overcomes this asymmetry: Whenever the LSB needs to be changed, the algorithm "flips a coin." If the obverse is up, the value is incremented, otherwise decremented. Although the change is never greater than one, this strategy can affect higher bits than just the LSB. For example 127 could be incremented to 128. This will change all eight bits.

PM1 embedding adjusts the least significant bit of carrier elements (e.g., pixel values) to match the message bit. Its result is compatible to LSB extraction, the embedded (and possibly encrypted) message is directly read from a sequence of least significant bits.

We looked at existing literature on PM1 detection and implemented some of those detectors. They performed less reliably than expected. Our initial goal to develop a targeted detector for PM1 seemed to move afar. We did not like to go the path to derive generic blind feature sets, but watched the development. The most promising feature set with a comprehensive evaluation was contributed by Giacomo Cancelli recently [2].

Histogram Characteristic Function Center of Mass (HCFCOM)

Harmsen and Pearlman noted that PM1 embedding acts as a low-pass filter for the histogram h_1 of the image. Therefore the high frequency power in the histograms of PM1 steganograms is reduced. In other words, the FFT of the histogram H_1, also referred to as the histogram characteristic function (HCF), is likely to be significantly affected by PM1 embedding. In fact, its center of mass (COM), defined as

$$c_1(H_1) = \frac{\sum_{k=0}^{127} k|H_1(k)|}{\sum_{k=0}^{127} |H_1(k)|},$$

will be shifted toward the origin. The symmetric part of the FFT ($128 \leq k \leq 255$) is ignored here. This attack can be used with the `swb` package.

```
> require(swb)
> cover <- read.png("lena.png")$grey
> stego <- pm1.embed(cover)
> hcfcom.attack(cover)
[1] 8.326196
> hcfcom.attack(stego)
[1] 8.943867
```

This approach can be extended to multidimensional signals, e.g., RGB images, by using a multidimensional FFT and computing a multidimensional COM. Experimental results [14] have shown that the HCF strategy performs better with RGB images than with greyscale images.

Ker [14] suggested that this difference in performance is due to a lack of sparsity in the histogram of greyscale images. To address this issue, Ker proposed using a two dimensional adjacency histogram $h_2(k, l)$, which tabulates how often each pixel intensity is observed next to another.

Amplitude of Local Extrema (ALE)

The neighbourhood in the two dimensional adjacency histogram $h_2(k, l)$ can be defined in four different ways: horizontal, vertical, main diagonal, and minor diagonal. [3]

Wavelet Absolute Moments (WAM)

One of the earlier feature sets that can be used for blind detection of PM1 embedding is called Wavelet Absolute Moments (WAM) [11]. These features can be extracted with the function `abs.moments.attack`.

```
> abs.moments.attack(stego)
Loading required package: rwt0
 [1] 0.174671834 0.054164606 0.023816770 0.013684569 0.009844786 0.008620941
 [7] 0.009000971 0.011008924 0.015468531 0.024383003 0.042012892 0.077243565
[13] 0.149049799 0.041256937 0.016352419 0.008503192 0.005523534 0.004302428
[19] 0.003876143 0.003922311 0.004359665 0.005234122 0.006699314 0.009043658
[25] 0.204717740 0.071622717 0.035275153 0.022848144 0.018702908 0.018731848
[31] 0.022321434 0.030907515 0.048688359 0.085451097 0.163569729 0.334627520
```

2.7.2 Model-Based Steganography

Sallee modelled the marginal distribution of the DCT coefficients in JPEG images by the generalised Cauchy distribution [18]. In contrast to LSB steganography, the pairs of values are not equalised with his model-based approach. Instead, the embedded message is adapted to the generalised Cauchy distribution of each AC DCT subband in the JPEG carrier file. This adaptation is implemented as arithmetic decoding. Arithmetic coding transforms unevenly distributed bitstreams into shorter, uniform ones. Conversely, the arithmetic decoding can take a uniformly distributed bitstream (the message to be embedded) to produce a bitstream that is adapted to given probabilities of 0 and 1 according to the present generalised Cauchy distribution. In case the chosen distribution fits to the JPEG file, the first order statistics is preserved after embedding the adapted bitstream into the LSBs of the coefficients. This procedure is known as **MB1** today.

One weak property of MB1 is that block artefacts increase with growing size of the payload. **MB2** was developed to overcome this weakness [19]. It embeds the message in the same way as MB1 does but offers only half the capacity of MB1 to the user. The other half of the nonzero DCT coefficients are reserved for blockiness reduction. Early assessment by Fridrich showed good security compared to MB1 [6]. In later analyses, however, the tables have been turned [21, 17].

For the sake of simplicity we focus on the deblocking function and our most powerful feature set here. The reader is directed to our IWDW07 paper [22] for further information about other attacks on MB2 that we developed during this project.

Our key innovations behind MB2 detection are:

1. a **gradient aware blockiness** measure,

2. a classification of **coefficient types** regarding the blockiness reduction, and
3. a collection of suitable adjustment and calibration methods.

Blockiness

To reduce the blockiness, Sallee used a simple blockiness measure based on the difference of pixels at block boundaries. In Fig. 2.5a these pixels are marked grey.

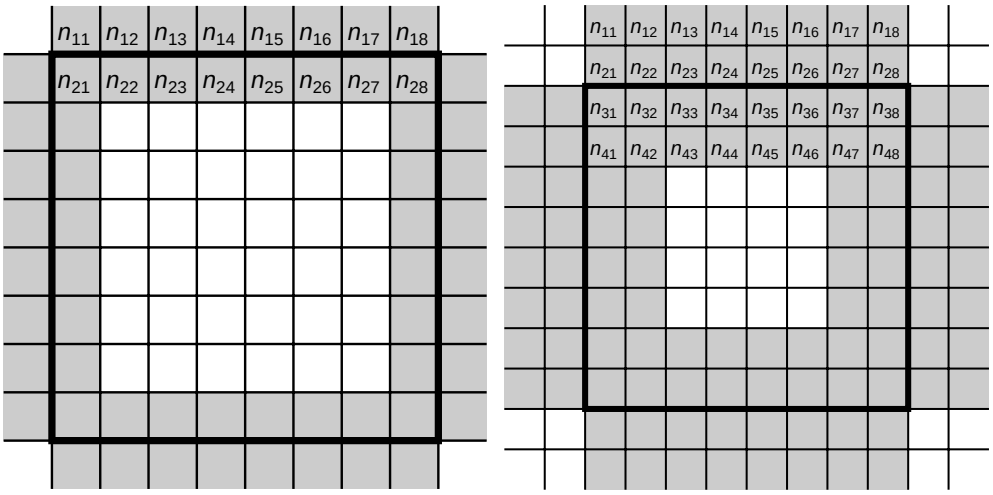


Figure 2.5: Pixels tested for (a) simple and (b) gradient aware blockiness

Each block has a north, west, south, and east boundary (with the pixels denoted by n_*, w_*, s_*, and e_*, respectively). The simple blockiness b_s is defined by the following sum of absolute differences:

$$b_s = \sum_{i=1}^{8} (|n_{1i} - n_{2i}| + |w_{1i} - w_{2i}| + |s_{1i} - s_{2i}| + |e_{1i} - e_{2i}|)$$

Corner pixels belong to several boundaries at the same time, e. g., $n_{28} = e_{21}$.

This simple blockiness measure is suitable as optimisation goal to be minimised if the gradients at the block boundary were initially flat. If there is a smooth gradient across the block boundary, even in the absence of a steganographic message, e.g. a transition from dark to light, then the blockiness reduction will equalise the brightness at the boundary. This would introduce a stairstep in brightness, which is not only a visible artefact but also detectable using a **gradient aware** blockiness measure.

We developed an improved measure that respects the gradient within the block while measuring the difference at the boundary [24]. Figure 2.5b shows more pixels (marked grey) that are considered to estimate the boundary gradient. The mean of the differences of the surrounding should be equal to the difference at the boundary

(2.2). This leads to the new, gradient corrected version of the blockiness measure b_g (2.5).

$$n_{2i} - n_{3i} \approx \frac{(n_{1i} - n_{2i}) + (n_{3i} - n_{4i})}{2} \tag{2.2}$$

$$0 \approx n_{1i} - 3n_{2i} + 3n_{3i} - n_{4i} \tag{2.3}$$

$$b_{g,n}^{(\lambda)} = \sum_{i=1}^{8} |n_{1i} - 3n_{2i} + 3n_{3i} - n_{4i}|^{\lambda} \tag{2.4}$$

$$b_g^{(\lambda)} = b_{g,n}^{(\lambda)} + b_{g,w}^{(\lambda)} + b_{g,s}^{(\lambda)} + b_{g,e}^{(\lambda)} \tag{2.5}$$

λ can be 1 or 2, depending on whether the absolute differences or a squared measure is desired.

Coefficient Types

The blockiness adjustment in MB2 leads to a differentiation of coefficients. The reason for this is that neither the coefficients used for embedding nor the coefficients that do not decrease the blockiness are altered during the blockiness adjustment. Regarding the blockiness reduction three sets of coefficient types can be defined. These three are the

fixed coefficients F**,** characterised by the fact that they cannot be altered because of model restriction even though they would decrease the blockiness if changed. The

different coefficients D are the ones that can be altered and if so they would decrease the blockiness. The remaining

indifferent coefficients I could be altered or not but if they are altered the blockiness would increase.

In order to categorise the coefficients a blockiness minimal image needs to be created. This is done with the function `deblock_delta` in the MB2 algorithm, which is also used for blockiness reduction. It is advisable to separate the coefficient types further into the disjoint sets DC, AC1 and AC0 of coefficients. **AC0 coefficients** are the AC coefficients that are zero. The separation is useful because MB1 and MB2 do not use DC and AC0 coefficients for embedding. Having the blockiness minimal image at hand the indifferent coefficients can be enumerated. Doing this we get indifferent AC1, DC and AC0 coefficients. To separate the different from the fixed coefficients we need to take a closer look at the restrictions of the blockiness adjustment. MB1 and MB2 only modify coefficients within their bin in order to keep the low precision bins unchanged so the receiver of the steganogram can calculate the same model of the image. Thus, if a coefficient value needed to be altered into another low precision bin in order to decrease the blockiness it is a fixed coefficient because it must not be altered that way to keep the extraction of the model parameters correct. This again could happen with DC, AC1 and AC0 coefficients. The remaining coefficients are the different ones. They

differ between the image and the blockiness minimal image but they could be altered without changing low precision bins and if so cause the blockiness to decrease.

In the case of MB1 a decrease of fixed and indifferent coefficients is expected while the different coefficients become more frequent. With MB2 the number of different coefficients should decrease because the longer the embedded message length the more coefficients are changed in order to decrease the blockiness, which can only be done by the different coefficients. Thus, the number of fixed coefficients increases, because different coefficients can only become indifferent or fixed. Since the blockiness is increasing during embedding the number of indifferent coefficients cannot increase significantly and so the fixed coefficients need to increase. Empirical analysis shows that the indifferent coefficients increase in most cases (92 %) too.

Adjustment and Calibration Methods

We apply the calibration method described by Fridrich et al. [9]: The JPEG image A is dequantised and decompressed to spatial domain, cropped by a margin of 4 pixels from the left and from the top, then recompressed to the calibrated JPEG image B using the quantisation table of A. The cropping shifts the DCT block raster by half a block size.

Since the size of the payload embedded into the image under test is unknown, we produce the image at full embedding rate. However, in most cases the full embedding rate causes the MB2 blockiness adjustment to fail. For this reason we embed only 95 % of the capacity with both, MB1 and MB2 (separately, not on top of MB1, yielding two JPEG images).

Feature Set

The most powerful feature set $\mathcal{F}$ that we developed includes five features, which we extract from six variants of the JPEG image. The resulting 30 features can be denoted as follows:

$$\mathcal{F} = \left\{ \begin{array}{c} b_{\mathrm{g}}^{(1)}, \\ b_{\mathrm{g}}^{(2)}, \\ |F_{\mathrm{AC0}}|, \\ |D|, \text{ and} \\ |I_{\mathrm{AC0}}| \end{array} \right\} \text{extracted from} \left\{ \begin{array}{c} A, \\ B, \\ A + 95\ \%\ \text{MB1 embedding}, \\ A + 95\ \%\ \text{MB2 embedding}, \\ B + 95\ \%\ \text{MB1 embedding, and} \\ B + 95\ \%\ \text{MB2 embedding}. \end{array} \right\}. \tag{2.6}$$

Chapter 3

Results and Discussion

3.1 Possible sources of error

A meaningful comparison with previous state of the art is nearly impossible if the results are validated only by a single experiment carried out on the famous “Lena” image and the algorithm is vaguely described. However, even if the algorithm is precisely described and all parameters are listed in a table, the validity still depends on the data set that is needed to obtain the same results of the original paper.

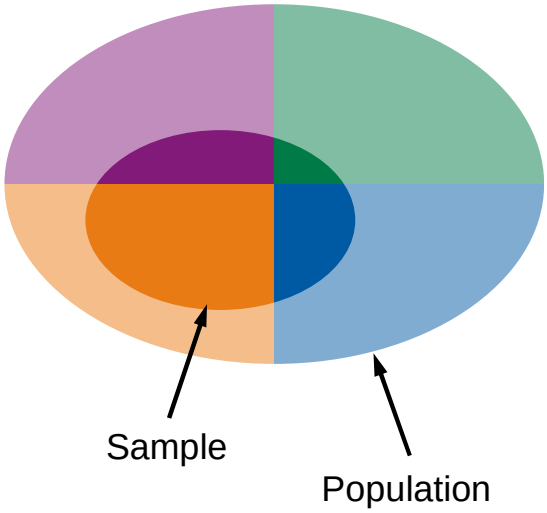


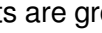
Figure 3.1: Coverage error of a sample, represented in a Venn diagram

A coverage error occurs when all members of the population do not have an equal (or known) probability of being included in the sample. If the sample is not carefully matched, the performance of a detector can considerably deviate in different evaluations.

Sampling error or estimation error is the error caused by observing a sample instead of the whole population, which is always the case in steganalysis. This type

of error decreases as the sample grows. High performance computing enables tests with larger image databases.

3.2 Spatial Domain

In the original version of RS, Fridrich et al. divided the image into "disjoint groups of n adjacent pixels $(x_1, \ldots, x_n)$." [8] The group size is variable. The masks that performed best in their experiments are groups of 4×1 pixels with the mask  as well as 2×2 pixels with the mask  .

Ker asked whether the groups should be disjoint or overlap [13]. In the similar case of Pairs Attack [10], where the homogeneity of colour cuts is measured, he argued conclusively that the groups of pairs must overlap so that every pair of adjacent pixels is considered. Also Dumitrescu used overlapping groups in their Sample Pairs Analysis (SPA) [4, 5]. However, Ker found lower reliability of RS for overlapping groups compared to their original disjoint version.

Ker experimented compared at least five linear masks and five square shaped masks [12]. He found that groups of 3×3 pixels and the mask  performed best, leading to a significant improvement compared to the standard mask [13]. Although we appreciate the way he set-up his image database, it was not possible for us to reproduce this improvement with our own image database. This might be a difference in implementation, since we got improved results for other 3×3 masks, but it could also be a property of our image database that consists only of never compressed images—a typical covering error problem (cf. Section 3.1).

Standard RS determines the estimated length of the embedded message by solving a quadratic equation based on the cardinalities of regular and singular groups. RS fails if this equation has no real solution. Ker introduced a different measure that completely ignores singular groups, but is a relative difference based on the terms of regular groups only. This significantly improves the reliability of RS. Ker observed that the new mask with the larger group size provides no improvement when combined with the relative difference measure.

Only a handful of all possible masks has been tested in the literature. We guess that in terms of performance there are cross-dependencies between mask pattern, type of length estimation (relative difference or standard length estimation), and whether groups are disjoint or overlap.

We used 10,000 original images from the BOWS-2 image database. These images were taken with a number of different digital cameras. The shorter edge of the images was scaled to 512 pixels, then the center 512×512 pixels were taken and converted to 8 bit greyscale. In contrast to the 10,000 images offered on the BOWS-2 webpage, the images used here do not contain any BOWS-2 watermark.

We considered 4 mask patterns with 2×2 pixels, 100 with 3×3 pixels, and 776 linear mask patterns with length $2 \ldots 10$ pixels. The linear masks were applied three times: after scanning row by row and along a Hilbert 1 and 2 path. The attack was applied in four variants:

1. standard length estimation and disjoint groups (original),

Table 3.1: 3×3 disjoint groups (a) with length estimation (b) with relative difference measure

(a)		
Rank	Mask	reliability
1		0.6789
2		0.6759
3		0.6751
4		0.6747
5		0.6709
6		0.6690
7		0.6659
8		0.6651
9		0.6597
10		0.6594
	...	
91		0.5486
92		0.5484
93		0.5362
94		0.5339
95		0.5323
96		0.5229
97		0.5168
98		0.4915
99		0.4737
100		0.4495

(b)		
Rank	Mask	reliability
1		0.7108
2		0.7027
3		0.7014
4		0.7002
5		0.6984
6		0.6984
7		0.6956
8		0.6941
9		0.6917
10		0.6898
	...	
91		0.5845
92		0.5808
93		0.5803
94		0.5734
95		0.5635
96		0.5588
97		0.5487
98		0.5102
99		0.5031
100		0.4880

2. standard length estimation and overlapping groups,
3. relative difference measure and disjoint groups, and
4. relative difference measure and overlapping groups.

Each image was attacked twice, before and after embedding a random message that uses 3 % of the capacity. This amounts to a total of almost **200 million RS attacks**, which will be presented in the following rank tables (Tables 3.1 . . . 3.7).

3.2.1 Results for 3×3 Groups

Table 3.1 considers 100 mask patterns with 3×3 pixels. All of the following rank tables present the first ten and last ten ranks. This ranking also depends on the image source and probably also on the actual embedded message. Table 3.1 presents the results for disjoint, non-overlapping groups. The attack seems to be more reliably for

"heavy" masks (more ones than zeros) than for "light." The reliability is between 1 (perfect separation) and 0 (when the area under the ROC curve is 0.5). The standard RS length estimation considers the cardinalities of both sets, the regular groups R and singular groups S to estimate the length. Since we are not interested in the precision, but only in the reliability in this case, we compare its performance to one of the relative difference $(|R_{-m}| - |R_m|)/(|R_{-m}| + |R_m|)$, where m is the mask used to determine if groups are regular.

The case for overlapping 3×3 mask patterns is shown in Table 3.2. Apart from

Table 3.2: 3×3 overlapping groups (a) with length estimation (b) with relative difference measure

(a)		
Rank	Mask	reliability
1		0.7729
2		0.7639
3		0.7637
4		0.7630
5		0.7620
6		0.7619
7		0.7614
8		0.7611
9		0.7610
10		0.7603
	...	
91		0.7374
92		0.7372
93		0.7364
94		0.7363
95		0.7316
96		0.7297
97		0.7258
98		0.7250
99		0.7072
100		0.7041

(b)		
Rank	Mask	reliability
1		0.7899
2		0.7851
3		0.7811
4		0.7798
5		0.7798
6		0.7790
7		0.7786
8		0.7783
9		0.7779
10		0.7772
	...	
91		0.7495
92		0.7484
93		0.7478
94		0.7426
95		0.7422
96		0.7405
97		0.7384
98		0.7246
99		0.7139
100		0.6693

the image margin, each pixel will be used in every position of the mask pattern here. We notice an improvement compared to the case of disjoint groups. The previous dependence between the weight of the mask and its reliability disappeared in the overlapping case. Also the difference between the two measures (standard RS length estimation; relative difference of R) is less distinctive. The mask selected by Ker [12] performed rather poor here. However, we verified the dominance of 3×3 masks, which performed best in our experiments.

3.2.2 Results for 2×2 Groups

There are only four 2×2 mask patterns (apart from isomorphic variants). Table 3.3 presents the ranking for their disjoint application, and Table 3.4 for overlapping. Again,

Table 3.3: 2×2 disjoint groups (a) with length estimation (b) with relative difference measure

(a)

Rank	Mask	reliability
1		0.6729
2		0.6532
3		0.6153
4		0.5692

(b)

Rank	Mask	reliability
1		0.6807
2		0.6590
3		0.6572
4		0.5802

Table 3.4: 2×2 overlapping groups (a) with length estimation (b) with relative difference measure

(a)

Rank	Mask	reliability
1		0.7493
2		0.7482
3		0.7080
4		0.7046

(b)

Rank	Mask	reliability
1		0.7354
2		0.7248
3		0.7184
4		0.6697

like in the 3×3 case, the attack works more reliably in the overlapping case. Also the dependence between heavy masks and stronger reliability could be true. However, while in the disjoint case the relative difference seems to be slightly more reliable compared to the standard RS length estimation, this is reversed in the overlapping case. Surprisingly, the heavy mask pattern  performed better than the proposed  by Fridrich et al. [8].

3.2.3 Results for Groups with Linear Masks

This part of experiments considered linear mask that are usually applied row by row. We also applied these masks along the two different versions of the Hilbert curve (cf. Fig. 2.2). Indeed, the optimum group size for our image set was slightly above 4 pixels, but below 9 pixels. However, the desired gain by using recursive scanpaths was not found. The usual row by row scan exceeded the reliability of Hilbert paths, regardless the masks are applied disjoint or overlapping, and regardless the measure was standard length estimation or relative difference of R. Table 3.5 shows the results

Table 3.5: Linear disjoint groups, scanned (a) row by row (b) along a Hilbert 1 curve (c) along a Hilbert 2 curve

(a)			(b)			(c)		
Rank	Mask	reliability	Rank	Mask	reliability	Rank	Mask	reliability
1		0.6956	1		0.6548	1		0.6790
2		0.6922	2		0.6527	2		0.6746
3		0.6879	3		0.6521	3		0.6744
4		0.6865	4		0.6511	4		0.6739
5		0.6860	5		0.6506	5		0.6680
6		0.6844	6		0.6492	6		0.6666
7		0.6833	7		0.6490	7		0.6666
8		0.6830	8		0.6456	8		0.6663
9		0.6828	9		0.6438	9		0.6647
10		0.6808	10		0.6433	10		0.6645
	...			...			...	
767		0.4913	767		0.4678	767		0.4790
768		0.4894	768		0.4659	768		0.4789
769		0.4871	769		0.4578	769		0.4752
770		0.4834	770		0.4565	770		0.4738
771		0.4760	771		0.4537	771		0.4734
772		0.4735	772		0.4441	772		0.4693
773		0.4500	773		0.4308	773		0.4229
774		0.4366	774		0.4181	774		0.4147
775		0.4255	775		0.4050	775		0.4126
776		0.3895	776		0.3879	776		0.4003

for the particular scanpaths, standard RS length estimation, and disjoint application of groups. Again we can see an advantage for heavy mask patterns. There is no significant difference between the favourite that was experimentally determined by Fridrich and the best reliability that we found in our experiments for linear masks and conventional row by row scanpath.

Table 3.6 presents the reliability for the relative difference measure of R. This is a minor improvement against the standard RS length estimation.

We yielded a much stronger gain when groups overlap. As mentioned earlier the other scanpaths performed worse and are omitted in Table 3.7. We noticed a marginal performance gain for the relative difference. Interestingly, there is a characteristic pattern at the top that includes a repeated sequence 0-1-0-1. Overlapping linear groups, scanned row by row, evaluated by relative difference yield the second best reliability, right after the 3×3 pixel masks.

Table 3.6: Linear groups with relative difference measure, scanned (a) row by row (b) along a Hilbert 1 curve (c) along a Hilbert 2 curve

(a)			(b)			(c)		
Rank	Mask	reliability	Rank	Mask	reliability	Rank	Mask	reliability
1		0.7139	1		0.6719	1		0.7068
2		0.7111	2		0.6704	2		0.7007
3		0.7093	3		0.6682	3		0.6934
4		0.7084	4		0.6681	4		0.6920
5		0.7083	5		0.6668	5		0.6911
6		0.7065	6		0.6663	6		0.6908
7		0.7064	7		0.6651	7		0.6897
8		0.7058	8		0.6638	8		0.6895
9		0.7056	9		0.6636	9		0.6881
10		0.7044	10		0.6635	10		0.6879
	...			...			...	
767		0.5297	767		0.5009	767		0.5176
768		0.5261	768		0.4971	768		0.5175
769		0.5212	769		0.4934	769		0.5150
770		0.5179	770		0.4923	770		0.5128
771		0.5157	771		0.4917	771		0.5069
772		0.5148	772		0.4799	772		0.5017
773		0.4931	773		0.4681	773		0.4659
774		0.4765	774		0.4559	774		0.4549
775		0.4679	775		0.4424	775		0.4524
776		0.4296	776		0.4228	776		0.4388

3.3 DCT Domain

3.3.1 Impact of Image Size

For the comparison in Fig. 3.2 we downloaded 630 large TIFF images (1500 $\times$ 2100 pixels) from the NRCS database [16]. These images have been downsized (using pnmscale) to five different sizes (600 $\times$ 840, 400 $\times$ 560, 200 $\times$ 280, 80 $\times$ 112, and 40 $\times$ 56 pixels), converted to greyscale, and JPEG compressed with quality $q = 0.8$. For the comparison in Fig. 3.2, we selected images with an embedding rate of 1 % of the capacity. In abundance of results we show only the best representative for each fundamental estimation method. This selection is based on the between-image error at an embedding rate of 1 % of the capacity (600 $\times$ 840, JPEG quality 0.8). The between-image error is related to the detection power, since this kind of error dominates at low embedding rates, while the within-image error is negligible and the overall bias in the estimation is without influence to reliability. The representatives are the JPairs attack [25] applied to AC coefficients only, scanned interblock along a Hilbert 1 path, the JSPA attack [25] applied to AC coefficients only, scanned interblock row by row, the attack by Zhang and Ping [29], the category attack by Lee et al. [15],

Table 3.7: Overlapping linear groups, scanned row by row (a) with length estimation (b) with relative difference measure

(a)		
Rank	Mask	reliability
1		0.7695
2		0.7689
3		0.7689
4		0.7689
5		0.7689
6		0.7689
7		0.7643
8		0.7643
9		0.7643
10		0.7643
	...	
767		0.7030
768		0.7023
769		0.7023
770		0.7023
771		0.7023
772		0.7022
773		0.7022
774		0.7022
775		0.7022
776		0.7022

(b)		
Rank	Mask	reliability
1		0.7732
2		0.7732
3		0.7732
4		0.7732
5		0.7732
6		0.7726
7		0.7726
8		0.7726
9		0.7726
10		0.7726
	...	
767		0.7090
768		0.7090
769		0.7072
770		0.7072
771		0.7072
772		0.7072
773		0.7072
774		0.7072
775		0.7072
776		0.7071

the JWS attack [25] applied to AC coefficients only, scanned interblock row by row, the WB attack [25] applied to both AC and DC coefficients, scanned interblock along a slalom path, the JRS attack [25] with mask $(0, 1)$ applied to AC coefficients only, scanned intrablock in zigzag order, and finally the attack by Yu et al. [27]. As expected, the between-image error decreases with increasing image size. While the between-image error of the JPairs attack is smallest for images larger than 200×280, there are more suitable candidates for thumbnail images (40×56), where WB attack, category attack, and the attack by Yu et al. do a better job. Interestingly, the JWS attack, which is closely related to the front runner WB, has the biggest problem with thumbnail images. One possible explanation could be the increased variance in downscaled images.

3.3.2 Impact of JPEG Quality

The medium sized images (200×280) have been compressed at seven different qualities ($q = 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99$). Again, we only selected images with an

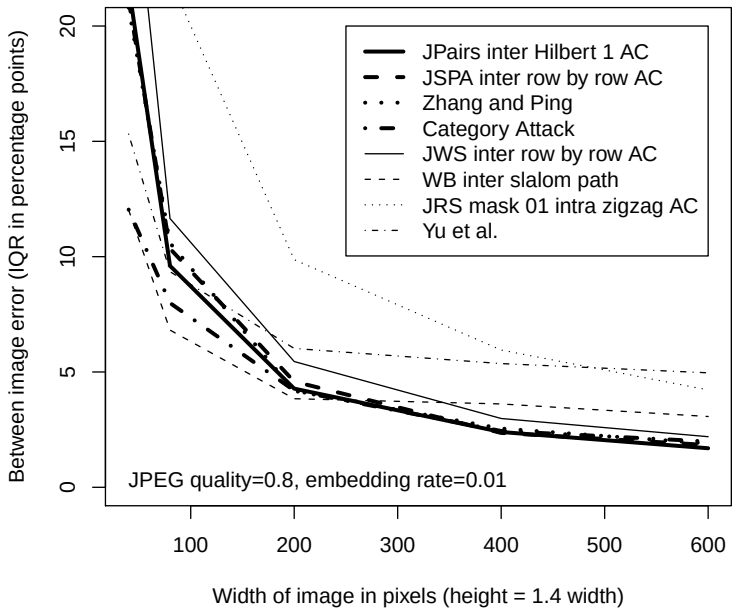


Figure 3.2: Between-image error as a function of the image size

embedding rate of 1 % of the image capacity here. Figure 3.3 shows the between-image error as a function of the JPEG quality. Except for the category attack and the attack by Zhang and Ping, this kind of error is increased for qualities above 0.9.

3.4 Steganalysis of Model-Based Steganography

For our experimental evaluation we downloaded 2300 large TIFF images (2100 $\times$ 1500 pixels) and 630 further images (1500 $\times$ 2100 pixels) from the NRCS database [16]. These images have been downsized to 840 $\times$ 600 pixels by pnmscale's default method, converted to greyscale, and JPEG compressed with quality $q = 0.8$. The rather small payload was 0.02 bits per nonzero coefficient for both, MB1 and MB2, using the original implementation by Phil Sallee.

The training was based on 2300 images and the classification was done by LDA based on 630 images. Table 3.8 shows the results, where "23" represents the attack by Fridrich with 23 DCT features [6], "274" the one with 274 mixed DCT and Markov features by Pevný and Fridrich [17], and "324" the set by Shi et al. [21] with the 324 Markov features. "81" and "193" is a segmentation of the 274 features, where "81" contains the calibrated averaged Markov features. $\mathcal{F}$ is the proposed feature set, defined in Eq. 2.6 on page 29.

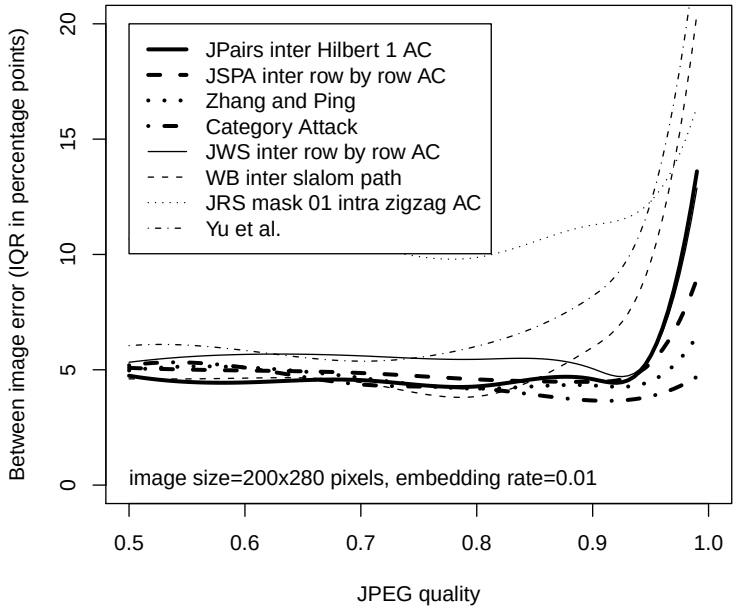


Figure 3.3: Between-image error as a function of the JPEG quality

Table 3.8: Detection reliability for feature combinations

		Additional features	Number of features				
			23	324	274	81	193
Reliability	MB1	—	0.181	0.597	0.698	0.585	0.516
		$\mathcal{F}$	0.596	0.793	0.791	0.743	0.708
	MB2	—	0.187	0.659	0.759	0.666	0.518
		$\mathcal{F}$	0.873	0.924	0.937	0.919	0.909
False alarm rate at 50 % detection	MB1	—	0.348	0.133	0.077	0.133	0.136
		$\mathcal{F}$	0.140	0.050	0.041	0.060	0.080
	MB2	—	0.370	0.105	0.042	0.086	0.147
		$\mathcal{F}$	0.019	0.008	0.006	0.011	0.011

Chapter 4

Conclusions

In our comprehensive study of the RS attack we found that there is no single mask pattern that works best in all cases. Instead we found inter-dependencies between all parameters, e.g. mask shape (squared or linear), application (disjoint or overlapping groups), and the measure used for detection (length estimate or relative difference). The surprising result is that not a single change of a particular parameter makes an appreciable difference. However, if we change several of them at once, the detection power of RS grows significantly for mask patterns that did not receive attention in the literature so far, despite the fact that LSB replacement in spatial domain images is probably the best understood steganalysis problem today.

We also contributed to the “harder” problems in steganalysis, namely the detection of model-based steganography MB2. Our proposed feature set is based on coefficient types that can be derived from the blockiness adjustment of MB2. Used in combination with existing blind feature sets the false positive rate is reduced from 10 % to 1 % for a very low embedding rate (0.02 bits per non-zero coefficient).

We developed a methodology to apply higher order steganalytic attacks from the spatial domain in the transformed domain. Based on 1700 million attacks, we evaluated the performance of the proposed attacks under diverse image parameters (size, quality), and determined the most advisable schemes (WB for small images, JPairs for larger ones).

The implementation of our steganographic workbench will almost certainly be extended during the next years. It will shorten the development and evaluation cycles for new attacks and new steganographic embedding methods.

References

[1] Rainer Böhme and Andrew D. Ker. A two-factor error model for quantitative steganalysis. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents VIII (Proc. of SPIE)*, volume 6072, pages 59–74, San Jose, CA, January 2006.

[2] Giacomo Cancelli. New techniques for steganography and steganalysis in the pixel domain, 2009. Ph.D.Thesis – Ciclo XXI.

[3] Giacomo Cancelli, Gwenaël Doërr, Ingemar J. Cox, and Mauro Barni. Detection of ± 1 LSB steganography based on the amplitude of histogram local extrema. In *IEEE International Conference on Image Processing ICIP 2008*, pages 1288–1291, San Diego, California, USA, October 2008.

[4] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang. Detection of LSB steganography via sample pair analysis. In Fabien A. P. Petitcolas, editor, *Information Hiding (5th International Workshop)*, volume 2578 of *LNCS*, pages 355–372, Berlin Heidelberg, 2003. Springer-Verlag.

[5] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang. Detection of LSB steganography via sample pair analysis. *IEEE Trans. of Signal Processing*, 51:1995–2007, 2003.

[6] Jessica Fridrich. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In Jessica Fridrich, editor, *Information Hiding (6th International Workshop)*, volume 3200 of *LNCS*, pages 67–81, Berlin Heidelberg, 2004. Springer-Verlag.

[7] Jessica Fridrich and Miroslav Goljan. On estimation of secret message length in LSB steganography in spatial domain. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents VI (Proc. of SPIE)*, San Jose, CA, 2004.

[8] Jessica Fridrich, Miroslav Goljan, and Rui Du. Detecting LSB steganography in color and grayscale images. *IEEE Multimedia*, 8(4):22–28, 2001.

[9] Jessica Fridrich, Miroslav Goljan, and Dorin Hogea. Steganalysis of JPEG images: Breaking the F5 algorithm. In Fabien A. P. Petitcolas, editor, *Information Hiding (5th International Workshop)*, volume 2578 of *LNCS*, pages 310–323, Berlin Heidelberg, 2003. Springer-Verlag.

[10] Jessica Fridrich, Miroslav Goljan, and David Soukal. Higher-order statistical steganalysis of palette images. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents V (Proc. of SPIE)*, pages 178–190, San Jose, CA, 2003.

[11] Miroslav Goljan, Jessica Fridrich, and Taras Holotyak. New blind steganalysis and its implications. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents VIII (Proc. of SPIE)*, volume 6072, pages 1–13, San Jose, CA, January 2006.

[12] Andrew Ker. Quantitative evaluation of Pairs and RS steganalysis. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents VII (Proc. of SPIE)*, pages 83–97, San Jose, CA, January, 19–22 2004.

[13] Andrew D. Ker. Improved detection of LSB steganography in grayscale images. In Jessica Fridrich, editor, *Information Hiding (6th International Workshop)*, volume 3200 of *LNCS*, pages 97–115, Berlin Heidelberg, 2004. Springer-Verlag.

[14] Andrew D. Ker. Steganalysis of LSB matching in grayscale images. *IEEE Signal Processing Letters*, 12:441–444, 2005.

[15] Kwangsoo Lee, Andreas Westfeld, and Sangjin Lee. Category Attack for LSB steganalysis of JPEG images. In Yun Qing Shi and Byeungwoo Jeon, editors, *Digital Watermarking (5th International Workshop) IWDW 2006 Jeju Island, Korea, November 8–10, 2006, Revised Papers*, volume 4283 of *LNCS*, pages 35–48, Berlin Heidelberg, 2006. Springer-Verlag.

[16] NRCS. Photo gallery of the USDA Natural Resources Conservation Service, 2006. Online available at `http://photogallery.nrcs.usda.gov/`.

[17] Tomáš Pevný and Jessica Fridrich. Merging Markov and DCT features for multi-class JPEG steganalysis. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents IX (Proc. of SPIE)*, San Jose, CA, January 2007.

[18] Phil Sallee. Model-based steganography. In Ton Kalker, Yong Man Ro, and Ingemar J. Cox, editors, *International Workshop on Digital Watermarking*, volume 2939 of *LNCS*, pages 154–167, Berlin Heidelberg, 2004. Springer-Verlag.

[19] Phil Sallee. Model-based methods for steganography and steganalysis. *International Journal of Image and Graphics*, 5:167–190, 2005.

[20] Dietmar Saupe. A unified approach to fractal curves and plants. In Heinz-Otto Peitgen and Dietmar Saupe, editors, *The Science of Fractal Images*, pages 273–286, New York NY, 1988. Springer-Verlag New York, Inc.

[21] Yun Qing Shi, Chunhua Chen, and Wen Chen. A Markov process based approach to effective attacking JPEG steganography. In Jan L. Camenisch, Christian S. Collberg, Neil F. Johnson, and Phil Sallee, editors, *Information Hiding (8th*

International Workshop), volume 4437 of *LNCS*, pages 249–264, Berlin Heidelberg, 2007. Springer-Verlag.

[22] Christian Ullerich and Andreas Westfeld. Weaknesses of mb2. In Yun Qing Shi, Hyoung-Joong Kim, and Stefan Katzenbeisser, editors, *Digital Watermarking (6th International Workshop) IWDW 2007 Guangzhou, China, December 2007, Proceedings*, volume 5041 of *LNCS*, pages 127–142, Berlin Heidelberg, 2008. Springer-Verlag.

[23] Andreas Westfeld. Space filling curves in steganalysis. In Edward J. Delp III and Ping Wah Wong, editors, *Security, Steganography and Watermarking of Multimedia Contents VII (Proc. of SPIE)*, pages 28–37, San Jose, CA, January, 16–20 2005.

[24] Andreas Westfeld. Lessons from the BOWS contest. In *Proc. of ACM Multimedia and Security Workshop 2006, MM&Sec06, Geneva, Switzerland*, pages 208–213, New York, September, 26–27 2006. ACM Press.

[25] Andreas Westfeld. Generic adoption of spatial steganalysis to transformed domain. In Kaushal Solanki, Kenneth Sullivan, and Upamanyu Madhow, editors, *Information Hiding. 10th Edition*, volume 5284 of *LNCS*, pages 161–177, Berlin Heidelberg, 2008. Springer-Verlag.

[26] Andreas Westfeld and Andreas Pfitzmann. Attacks on steganographic systems. In Andreas Pfitzmann, editor, *Information Hiding (3rd International Workshop)*, volume 1768 of *LNCS*, pages 61–76, Berlin Heidelberg, 2000. Springer-Verlag.

[27] Xiaoyi Yu, Yunhong Wang, and Tieniu Tan. On estimation of secret message length in Jsteg-like steganography. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)*, pages 673–676, 2004.

[28] Tao Zhang and Xijian Ping. A fast and effective steganalytic technique against Jsteg-like algorithms. In *Proceedings of the 2003 ACM Symposium on Applied Computing (SAC2003), March 9–12, 2003, Melbourne, Florida, USA*, pages 307–311, New York, 2003. ACM Press.

[29] Tao Zhang and Xijian Ping. A new approach to reliable detection of LSB steganography in natural images. *Signal Processing*, 83:2085–2093, 2003.

List of Symbols, Abbreviations, and Acronyms

AC	alternating current (DCT subbands with nonzero video frequency)
ALE	amplitude of local extrema, a feature set
BEST	Better Steganalysis, project title
BEST2	Better Steganalysis, phase 2
BIE	between image error
CA	Category Attack
COM	center of mass
CPU	Central Processing Unit
DC	direct current (DCT subbands with zero video frequency)
DCT	discrete cosine transform
FFT	fast Fourier transform
GPL	General Public Licence
HCF	histogram characteristic function
IQR	inter quartile range
JPairs	Pairs attack adopted to JPEG domain
JPEG	Joint Photographic Experts Group (lossy image compression format)
JRS	RS attack adopted to JPEG domain
JSPA	SPA adopted to JPEG domain
JWS	WS attack adopted to JPEG domain
LSB	Least Significant Bit
LTSB	Least Two Significant Bits
MB1	Model-based steganography (initial variant)
MB2	Model-based steganography (with deblocking)
Pairs	Pairs attack
PM*k*	Plus-Minus-*k* embedding
PNG	Portable Network Graphics (lossless image compression format)
ROC	Receiver Operating Characteristic
RS	An attack based on *regular* and *singular* groups of pixels

SPA	Sample Pairs analysis attack by Dumitrescu
sqrt	square root
WAM	wavelet absolute moments, a feature set
WB	weighted non-steganographic boundary attack
WIE	within image error
WPC	Wet Paper Codes
WS	An attack based on an estimation weighted statistics from surrounding pixels by Fridrich and Goljan
Yu	attack to Jsteg-like embedding using modified Cauchy model fitting by Yu et al.
ZP	simple attack by Zhang and Ping